CPSC 1155 – Assignment 5

C++ Functions

Objectives

The goal of this assignment is to write functions in C++.

Readings

You should be reading Chapter 6 of the textbook. The lectures and labs will provide additional supporting material.

Instructions

For each **Problem Statement**, follow the steps below:

1. Read the problem statement and clarify the problem.
    a. Break the problem into smaller problems if needed.
2. Determine the IPO.
    a. Determine input, output, intermediate variables, constants, conditions, and repetitions.
    b. Declare the variables and constants (data type + meaningful names).
    c. Work out the problem by hand using typical input values. Determine the range of valid input values.
    d. Determine the process.
3. Write a function as required. For each function:
    a. Determine the parameters and their data types.
    b. Determine the data type for the returned value.
4. Write the test program (main) as required. Reading input values and displaying results happen in the test program.
    a. Add comments where needed. Make sure to use a comments header to reflect the intention of your program and name of the author (you) and the date the program was written.
    b. Test, debug, and execute the program using typical values.

Submit according to the instruction in the "Submission" section.

Problem Statements

1. (temp.cpp) Write **two functions** that implement the following conversions:
    c. Function *celsius* receives a Fahrenheit temperature and returns its Celsius equivalent.
    d. Function *fahrenheit* receives a Celsius temperature and returns its Fahrenheit equivalent.

    Write a test program that uses *celsius* and *fahrenhiet* to print the following table.

```
Celsius   Fahrenheit   |  Fahrenheit   Celsius
40.0      104.0        |  120.0        48.89
39.0      102.2        |  110.0        43.33
...
32.0      89.6         |  40.0         4.44
31.0      87.8         |  30.0         -1.11
```

2. (display_char.cpp) Write a **function** using the following header:

```
void printChars(char ch1, char ch2, int numberPerLine)
```

This function prints the characters between ch1 and ch2 inclusive with the specified number of characters per line (numberPerLine). Assume ch1 and ch2 are both uppercase or lowercase letters.

Write a test program that reads two characters and the number of characters per line, uses *printChars* to print the characters (pass the smaller one as the first character) with the required format. Characters are separated by exactly one space.

```
Enter first character: B
Enter second character: H
Enter number per line: 3
The characters are:
B C D
E F G
H
```

3. (sum_series.cpp) Write a **function** called *sumSeries* that receives i as parameter and computes m(i) with the following summation:

$$m(i) = \frac{1}{2} + \frac{2}{3} + \cdots + \frac{i}{i+1}$$

Write a test program that uses *sumSeries* to display the following table:

```
i       m(i)
1       0.5000
2       1.1667
…
19      16.4023
20      17.3546
```

4. (prime.cpp) An integer is said to be prime if it is divisible by only 1 and itself. For example, 2, 3, 5 and 7 are prime, but 4, 6, 8 and 9 are not.

Write a **function** called *isPrime* that receives an integer and determines whether the integer is prime or not.

Write a test program that uses *isPrime* to determine and prints all the prime numbers between 1 and 1000. Display 10 numbers per line.

5. (matrix.cpp) Write two **functions** using the following header:

```
int generate01()
void printMatrix(int n)
```

The first function generates and returns a random number either 0 or 1.

The second function prints an n-by-n matrix. Each element of the matrix is using the number generated by the first function.

Write a test program that prompts the user to enter n and uses *printMatrix* to display an n-by-n matrix. Here is a sample run:

```
Enter n: 3
0  1  0
0  0  1
```

```
0  0  0
```

6. (palindrome.cpp) Write a **function** named *reverse* that receives a string and returns its reverse.

   Write another **function** called *reverse* (function overload) that receives an integer and returns its reverse.

   Write two more overloaded **functions** called *isPalindrome*. One of these functions receives a string and uses *reverse* to determine if the string is palindrome or not. The other function does the same for an integer.

   Write a test program that prompts the user to first enter a string and then a positive integer and uses the above functions to determine if each of them is palindrome or not and then display the result.

7. (fibo.cpp) The Fibonacci series:

   0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, …

   begins with the terms 0 and 1 and has the property that each succeeding term is the sum of the two preceding terms.

   Write a (nonrecursive) **function** called *fibonacci* that receives an integer n and calculates and return the nth Fibonacci number. Use unsigned int for the function's parameter and unsigned long long int for its return type. For example, if the function received 8, it would return the 8th Fibonacci number 13.

   Write a test program that prompts the user to enter an input n and uses *fibonacci* to display the Fibonacci series from the 1st to nth.

   What is the maximum input that you could enter? Look for the maximum input and display it (Hint: if you enter an input larger than maximum input, the Fibonacci series would become incorrect at some point):

   ```
   Enter a number: 9
   The series is: 0 1 1 2 3 5 8 13 21
   The maximum input is your answer
   ```

8. (replace_vowel.cpp) Write a **function** called *isVowel* that receives one parameter as character and returns true if it is a vowel (A, E, I, O, U) letter. Write another **function** called *replaceV* that receives two parameters as string (parameter1) and character (parameter2) and returns a new string that replaces all the vowel letters from parameter1 with parameter2. The *replaceV* function should use *isVowel* function to check each character in the string. Make sure functions work for both uppercase and lowercase letters.

   Write a test program that reads a string and a character and uses *replaceV* to get the new string then displays the new string. Here is a sample run:

   ```
   Enter a String: Oh Hello world!
   Enter a character: $
   The new string is: $h H$ll$ w$rld!
   ```

Submission

Submit a zip folder named as yourName_Assign5.zip to Brightspace. This folder should consist of the C++ codes in individual .cpp files. Please name each cpp file exactly as the instructed names at the beginning of each question.

Please make sure that all your .cpp files compile and run properly before submission. Your file must run properly in order to receive full marks.

Marking Scheme

There are 10 marks for each question with the following details:

Question 1: temp.cpp

- 7 for two correct functions
- 3 for correct display result

Question 2: display_char.cpp

- 10 for correct solution

Question 3: sum_series.cpp

- 7 correct function
- 3 for correct display result

Question 4: prime.cpp

- 7 for correct function
- 3 for correct display result

Question 5: matrix.cpp

- 7 for two correct functions
- 3 for correct display result

Question 6: palindrome.cpp

- 10 for correct solution

Question 7: fibo.cpp

- 4 for correct function
- 3 for correct display result
- 3 for the correct answer

Question 8: replace_vowel.cpp

- 10 for correct solution